Lambda Functions: Takeaways ⤶

Syntax

- Accessing the first character in a string:

  ```
  s = "string"[0]
  ```

- Accessing the first four characters of a string:

  ```
  stri = "string"[0:4]
  ```

- Omitting the ending index to access the last five characters of a string:

  ```
  sword = "password"[3:]
  ```

- Skipping indexes in a slice:

  ```
  hlo = "hello world"[:5:2]
  ```

- Stepping backwards in a string:

  ```
  olleh = "hello world"[4::-1]
  ```

- Applying a function to a list of items:

  ```
  list(map(func, my_list))
  ```

- Filtering through a list of items using a function:

  ```
  def is_palindrome(my_string):

      return my_string == my_string[::-1]

  palindrome_passwords = list(filter(is_palindrome, passwords)))
  ```

- Using a lambda function to filter a list:

  ```
  numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

  evens = list(filter(lambda x : x % 2 == 0, numbers))
  ```

Concepts

- We can slice or select fragments of string objects using the index of the string. Slicing a string extracts a chunk or substring from the original string. You can either slice by specifying a single number to select a specific character or a range of values to select multiple characters.

- When extracting the first four characters in a string, you would specify the starting index as `0` but the ending index as `4`. Python uses the ending index to stop iterating and doesn't return the character at the ending index.

- Python's flexibility with ranges allow you to omit the starting or ending index to extract strings.

- Lambda functions, or anonymous functions, are used when you want to run a function once and don't need to save it for reuse.

Resources

- [Documentation for map, filter, and reduce](#)
- [Anonymous function](#)